LAPIN AMK
Lapland University of Applied Sciences

KISSAN RUOKINTASOVELLUKSEN KEHITTÄMINEN ANDROIDILLE

Henna Olli

Opinnäytetyö
Tekniikka ja liikenne
Tietotekniikka
Insinööri (AMK)

2015

LAPIN AMK
Lapland University of Applied Sciences

Opinnäytetyön tiivistelmä

Tekniikka ja liikenne
Tietotekniikka

Tekijä	Henna Olli	Vuosi	2015
Ohjaaja	Maisa Mielikäinen		
Toimeksiantaja	Lapin ammattikorkeakoulu		
Työn nimi	Kissan ruokintasovelluksen kehittäminen Androidille		
Sivu- ja liitemäärä	39 + 0		

Yhä useammat kissanomistajat haluavat tarjota lemmikeillensä luonnonmukaisen raakaruokavalion. Tämä johtuu osittain siitä, että teolliset ruoat saattavat sisältää kissoille haitallisia raaka-aineita. Raakaruokinnan aloittaminen voi kuitenkin olla haastavaa ja työlästä, sillä se edellyttää perehtymistä niin tarjotun ravinnon sisältämiin ravintoaineisiin kuin kissan ravintoainetarpeisiin. Tämän opinnäytetyön tavoitteena oli kehittää työkalu helpottamaan raakaruokavalion suunnittelua Android-sovelluksen muodossa.

Sovelluksen kehitykseen sovellettiin prototypoivan ohjelmistokehityksen mallia. Kehityksessä käytettiin Android Studiota. Raportissa käsitellään sovelluksen kehityksen vaiheiden lisäksi Androidia ja muita käytettyjä tekniikoita yleisellä tasolla.

Työn tuloksena syntyi sovellus, jolla voi suunnitella kissan raakaruokavaliota. Sovellus näyttää suunnitellun ruokavalion sisältämät ravintoaineet ja kissan yksilölliset ravintoainetarpeet visuaalisessa muodossa, jolloin niitä voi helposti verrata toisiinsa. Sovelluksen kehitys jatkuu ja sen tarjoamia ravintoainetietoja tullaan tarkentamaan.

Avainsanat android, ohjelmistokehitys, kissan raakaruokinta

LAPIN AMK
Lapland University of Applied Sciences

Abstract of Thesis

Technology, Communication and Transport
Degree Programme in Information Technology

Author	Henna Olli	Year	2015
Supervisor(s)	Maisa Mielikäinen		
Commissioned by	Lapland University of Applied Sciences		
Subject of thesis	Development of a Cat Feeding Application for Android		
Number of pages	39 + 0		

More and more cat owners want to provide natural raw diets to their pets due to the fact that commercial foods may contain ingredients that are harmful to cats. However, beginning raw feeding can be challenging and time consuming, as it requires a thorough understanding of both the nutrients contained in the food and the cat's nutritional needs. The aim of this study was to develop a tool, in the form of an Android application, to facilitate the planning of a raw diet.

A prototype-driven software development model was utilized during the development of the application. The application was developed using Android Studio. In addition to the stages of the development process, this report discusses Android and the other technologies used at a general level.

The study resulted in an application that can be used to plan the raw diet of a cat. The application displays the nutrients contained in the planned diet as well as the cat's nutrition needs in a visual form, making them easy to compare. Development of the application will continue, and the accuracy of the nutritional information it provides will be further improved.

Key words android, software development, raw feeding of cats

SISÄLTÖ

KÄYTETYT MERKIT JA LYHENTEET

Activity	Android-sovelluskomponentti. Tarjoaa näkymän, jonka kanssa käyttäjä on vuorovaikutuksessa (Android 2015)
Android	mobiilikäyttöjärjestelmä
API-taso	viittaa Androidin ohjelmistokehyksen versioon
autentikaatio	käyttäjän identiteetin varmennus
automaatiotestaus	testauksen suorittaminen automaattisesti
AVD	Android Virtual Device, virtuaalinen Android-laite
Broadcast receiver	Android-sovelluskomponentti, joka vastaa järjestelmän laajuisiin lähetyksiin (Android 2015)
CDM	Conceptual Data Model, konseptitason tietomalli
Content provider	Android-sovelluskomponentti, joka hallinnoi sovellusten yhteistä dataa
debuggaus	ohjelmakoodin virheiden paikallistamista ja korjaamista
Fragment	Android-komponentti, Activity pienoiskoossa (Android 2015)
instanssi	esiintymä, ilmentymä, tapaus
iteraatio	ohjelmiston kehityskierros
Kanban	visuaalinen työnhallintametodi (Haikala & Mikkonen 2011, 55)
Manifest	Viittaa AndroidManifest.xml-tiedostoon. Tiedoston avulla määritellään sovelluksen piirteet Android-ohjelmistoympäristölle (Xamarin Inc 2015)
POJO	Plain Old Java Object, Java-luokka, jonka ainoa tehtävä on toimia määrittelemänsä olion mallina (Webopedia 2014)
perusavain	tietokantataulun sarake, joka yksilöi taulun rivit (Hovi, Huotari & Lahdenmäki 2005, 343)
raakaruokinta	kissojen ja koirien luonnonmukainen ruokintatapa (YSA 2010)
refaktorointi	lähdekoodin rakenteen ja luettavuuden parantamista muuttamatta koodin rakennetta (Itkonen 2012)
relaatiotietokanta	relaatiomallia noudattava tietokanta (Hovi ym. 2005, 344)

SAP	saksalainen ohjelmistovalmistaja
SDK	Software Development Kit, ohjelmiston kehitystyökalut
sekvenssikaavio	kuvaa olioiden välisiä operaatioita
Service	Android-sovelluskomponentti, prosessi, joka toimii sovelluksen taustalla (Android 2015)
SQL	Structured Query Language, tietokantakieli
SQLite	Androidin täysin tukema kevyt relaatiotietokanta
UI	User Interface, käyttöliittymä
UX	User Experience, käyttäjäkokemus
XML	Extensible Markup Language, tekstimuotoisen tiedon merkintätapa (W3C 2015)

KUVIO- JA KOODIESIMERKKILUETTELO

1 JOHDANTO

Nykyään yhä useammat kissanomistajat haluavat pitää mahdollisimman hyvää huolta lemmikeistään. Kunnollinen ruokavalio on lemmikkien hyvinvoinnin perusta ja yksi tapa tarjota sellainen on raakaruokinta. Monet ruoka- ja eläinkauppojenkin tarjoamista teollisista kissanruuista sisältävät raaka-aineita, esimerkiksi viljaa ja maissia, jotka eivät kuulu kissojen luonnolliseen ruokavalioon ja voivat erityisesti pitkällä aikavälillä olla vaarallisia niiden terveydelle.

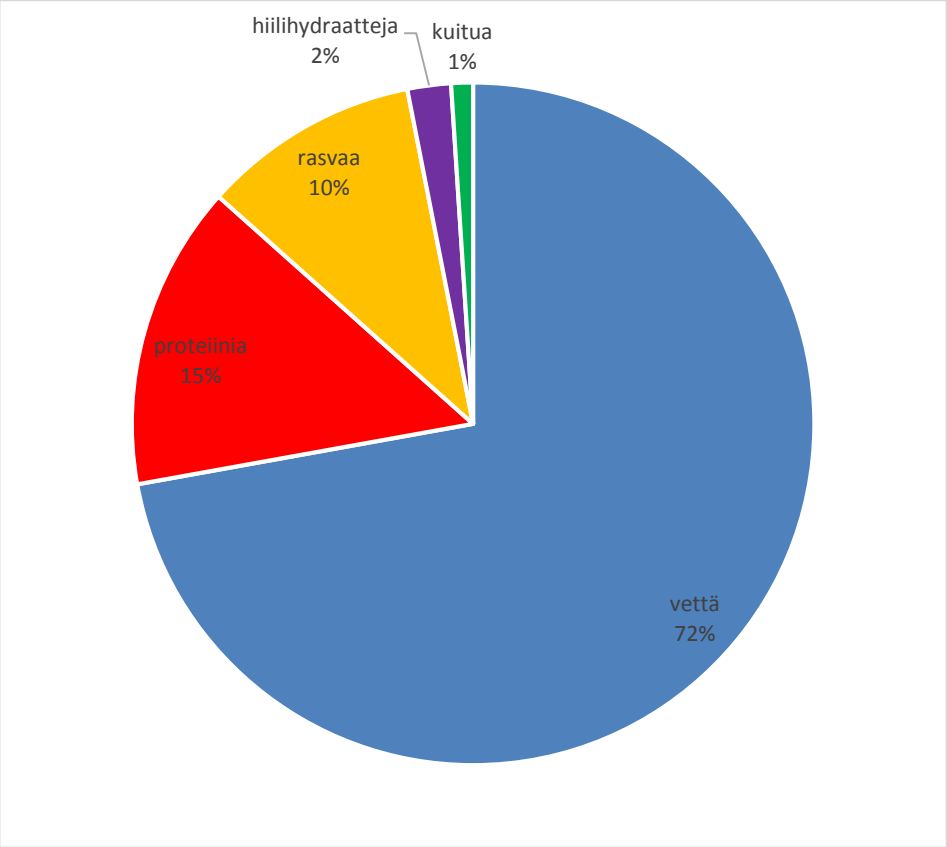
Ensimmäisen kissan luovutusikään tuloa odotellessa opiskeltiin kissojen ravintotarpeita ja huomattiin, että ruokalistojen tekeminen on todella työlästä. On paljon asioita, jotka tulee ottaa huomioon, mm. sopiva energiamäärä, riittävä vitamiinien saanti ja samalla yliannostuksen välttäminen, kalsium-fosforisuhde ja omega-rasvahappojen keskinäiset suhteet. Väärin toteutettuna raakaruokavaliokin voi olla kissalle haitallinen ja erityisesti sen aloittaminen on hankalaa valtavan tietomäärän vuoksi.

Hankaluuksista syntyi idea toteuttaa sovellus, joka helpottaa raakaruokinnan aloittamista ja suunnittelua ja kenties siten edistää kissojen hyvinvointia. Sovelluksen alustaksi valittiin älypuhelimet, koska niitä käytetään nykyään laajalti. Jotta sovellus tavoittaisi mahdollisimman monta kissanomistajaa, valittiin käyttöjärjestelmäksi tällä hetkellä markkinoita johtava Android.

Opinnäytetyön tavoitteeksi kehittyi siten suunnitella ja toteuttaa Android-sovellusmuotoinen työkalu kissan raakaruokavalion suunnittelua helpottamaan. Raportissa käsitellään aluksi kissan ravitsemusta ja Android-alustan ominaisuuksia. Sen jälkeen kuvataan prototypoivaa ohjelmistokehitysmallia soveltaen toteutetun sovelluksen kehitysvaiheet. Lopuksi esitetään työn tulos ja pohditaan siitä syntyneitä jatkokehitysmahdollisuuksia.

2 KISSAN RAVITSEMUS

Kissat ovat lihansyöjiä. Luonnossa kissat syövät enimmäkseen pieniä saaliseläimiä, kuten lintuja ja jyrsijöitä. Yksi tällaisista jyrsijöistä on hiiri, jonka koostumus on kuvattu kuvion 1 ympyräkaaviossa. Hiiren koostumuksesta voidaan päätellä ravintoarvosuhteet, joihin kissan elimistö on tottunut. Hiiressä on suuri määrä proteiinia, jonkin verran rasvaa ja minimaalinen määrä hiilihydraatteja. Kissa saa lisäksi pienen määrän kuitua, joka on peräisin hiiren ruoansulatuselimistöön jääneestä kasviperäisestä ravinnosta. (Zoran 2002; Lipponen 2012.)

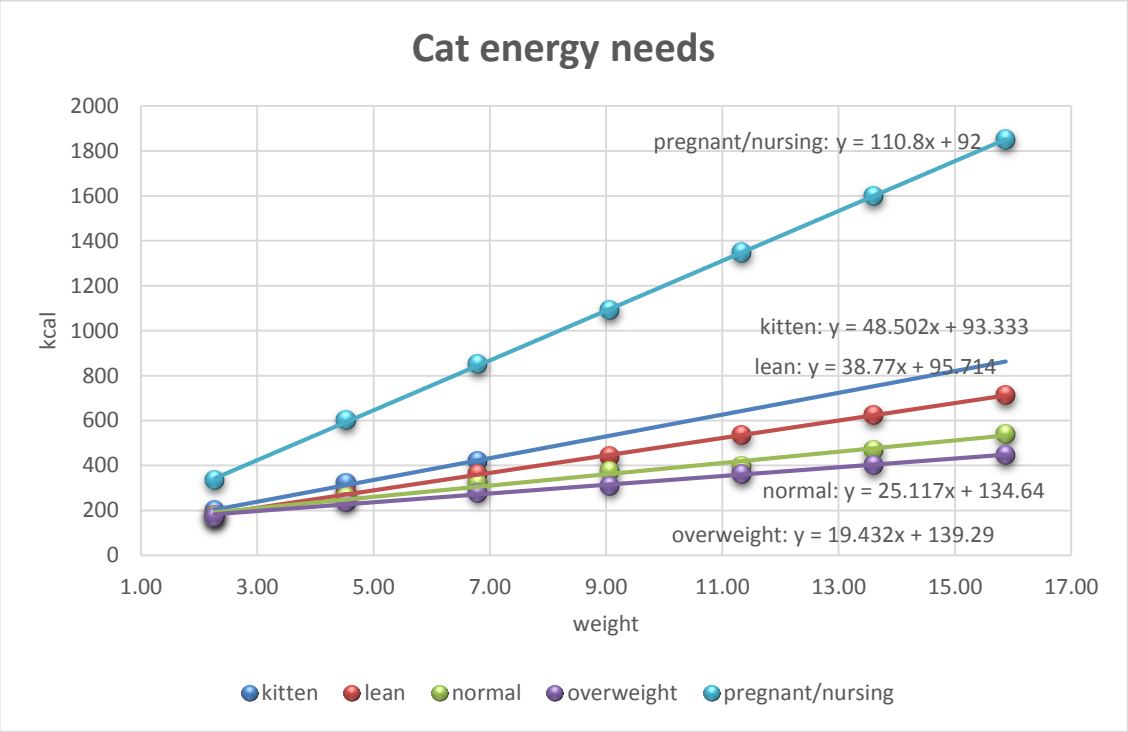


Kuvio 1. Hiiren ravintosisältö (Antell 2007)

Nykyään monet kissat asuvat sisätiloissa kuten niiden kannattaakin, mutta se tarkoittaa myös sitä, etteivät ne pääse hankkimaan ravintoaan luonnosta, vaan luottavat ravinnonsaannissa omistajaansa. Markkinoilla on kuitenkin pääasiassa tarjolla teollisia, prosessoituja ruokia, jotka eivät kuulu kissan luonnolliseen ruokavalioon. Ne sisältävät usein viljaa, vihanneksia ja muita kasviperäisiä raaka-aineita, joita kissa ei hyödynnä ihmisen tavoin ja jotka aiheuttavat ylipainoa ja dia-

betesta (Multanen 2013). Lisäksi tarjolla oleva kuivaruoka aiheuttaa helposti kissan kuivumisen ja johtaa virtsakiviin, koska alun perin kuivan alueen eläimenä kissa ei ole erityisen hyvä juomaan vettä ja on tottunut saamaan tarvitsemansa kosteuden ravinnostaan (Pierson 2013).

Kissojen onneksi raakaruokinta yleistyy jatkuvasti. Raakaruokinta, eli lemmikin ruokinta luonnonmukaisesti raa'alla lihalla ja luilla, on hyvä tapa vastata kissan uniikkeihin ravintoainetarpeisiin (Nederhoff 2012). Raakaruokinnassa on kuitenkin haasteensa, sillä väärin tehtynä sekin voi olla haitallista kissalle. Kissan tarpeisiin vastaavan ruokavalion suunnittelu voi olla erityisesti aloittelevalle raakaruokkijalle hankalaa, koska se vaatii paljon asioiden tutkintaa ja laskentaa. Toisin kuin teollisessa kissanruokapurkissa, ei broilerin jauhelihapakkauksessa kerrota tarkkaa ravintosisältöä. Opinnäytetyön tavoite on sen vuoksi saada aikaan työkalu ruokavalion suunnittelun helpottamiseksi.



Kuvio 2. Kissan energiantarve kunnon ja elämäntilanteen mukaan

Sovellusta varten koottiin ja analysoitiin tietoja kissan ravintoaine- ja energiantarpeista. Tietojen perusteella laskettiin optimaaliset hivenainetarpeet per painokilo, sekä proteiini-rasvasuhteet kasvaville ja aikuisille kissoille. National Research

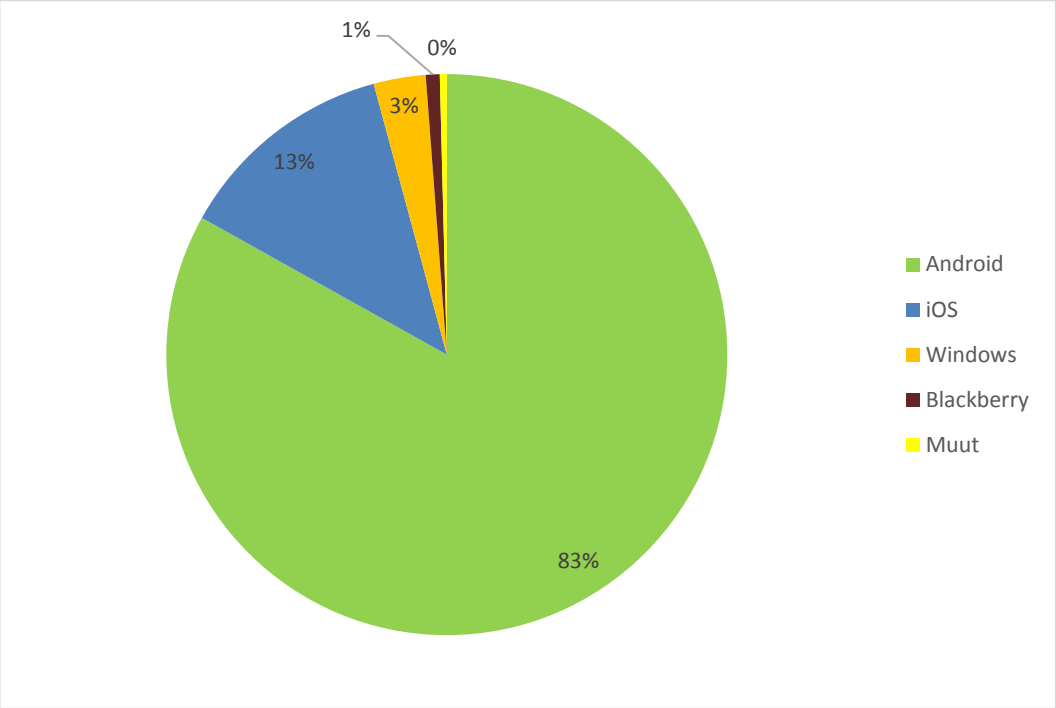
Council (2006) on laatinut energiantarvesuositukset kissoille, jotka painavat enintään yhdeksän kiloa. Suositusten pohjalta tehtiin kuvion 2 mukainen pistekaavio, jossa yhdeksää kiloa painavampien kissojen puuttuvat suositukset on määritelty lineaarisen ennusteen avulla. Lineaarisella regressioanalyysillä määriteltiin myös yhtälö kullekin kissatyypille. Esimerkiksi ylipainoisen kissan tapauksessa se on y = 35x +1, jossa x on kissan paino ja y ilmoittaa kalorintarpeen.

Laskettu energiantarve ei ole ehdottoman tarkka, koska kissat ovat kaikki yksilöitä yksilöllisillä tarpeilla. Toiset rodut esimerkiksi kasvavat pitempään kuin toiset ja pentueellisen kissan energiantarpeeseen vaikuttaa pentueen koko. Sen vuoksi raakaruokinnassa on numeroiden lisäksi tärkeää tarkkailla kissan yleiskuntoa ja painon kehitystä ja tehdä muutoksia ruokavalioon niiden perusteella.

3 ANDROID

3.1 Android-käyttöjärjestelmä

Android on Googlen perustaman Open Handsets Alliance -ryhmän kehittämä avoimen lähdekoodin ilmainen mobiilikäyttöjärjestelmä. Ryhmän tavoite oli saada aikaan käyttöjärjestelmä, joka on halvempi ja tarjoaa rikkaamman käyttökokemuksen ja kehitysmahdollisuudet kuin muut mobiilikäyttöjärjestelmät (Open Handset Alliance). Android SDK:n eli kehitystyökalujen versio 1.0 julkaistiin vuonna 2008 ja pian sen jälkeen markkinoille tuli ensimmäinen Androidilla varustettu älypuhelin (Google 2008).



Kuvio 3. Mobiilikäyttöjärjestelmien markkinaosuudet vuoden 2013 lopussa (Gartner 2014)

Ensijulkaisustaan lähtien Androidin suosio on kasvanut reilusti niin käyttäjien kuin kehittäjienkin keskuudessa. Kuviossa 3 esitetään eri mobiilikäyttöjärjestelmillä varustettujen älypuhelinten maailmanlaajuiset markkinaosuudet vuoden 2013 loppupuolella. Kuviosta voidaan havaita, että Androidin markkinaosuus on huomattavasti suurempi kuin sen kilpailijoiden, joista suurimmat ovat Applen iOS ja Microsoftin Windows Phone.

Androidin pinomuotoisen arkkitehtuurin pohjalla on Linux-ydin, jonka päällä toimii Dalvik-virtuaalikone. Jokainen Android-sovellus ajetaan omana instanssinaan virtuaalikoneesta; instanssit taas ajetaan Linux-ytimen prosesseina. Pinon seuraavalla tasolla on Androidin laaja ohjelmistokehys, jota niin kehittäjät kuin älypuhelinten valmistajat saavat käyttää vapaasti hyödykseen, kunhan pinon alimmat tasot jätetään koskemattomiksi. (Liu & Yu 2011; Skogberg 2010.)

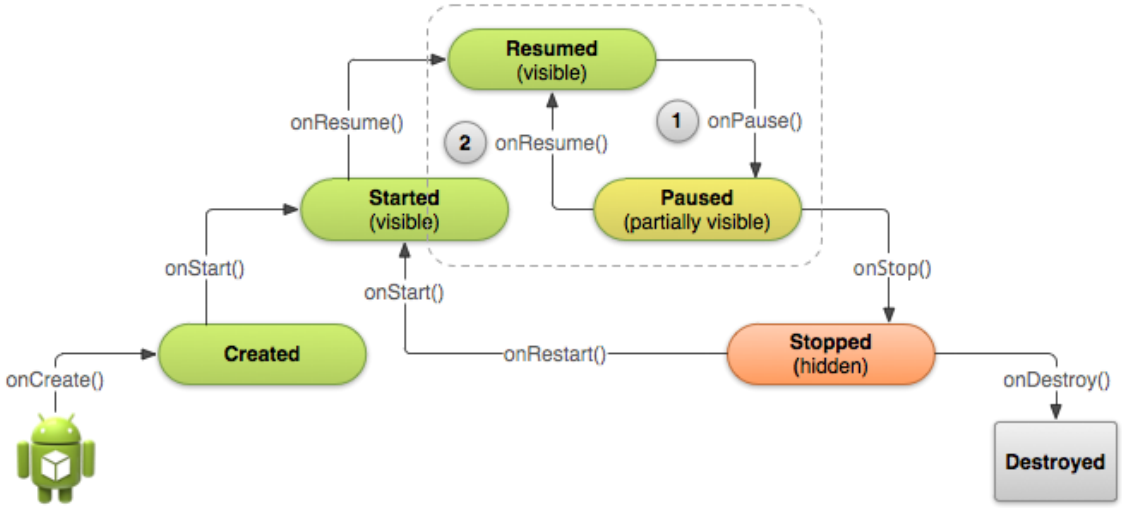
3.2 Android-sovelluskehitys

Google tarjoaa Android-sovelluskehitykseen ilmaiset työkalut Android SDK:n muodossa. Sovelluskehitykseen on kaksi virallista ohjelmointiympäristöä: Eclipse Android Development Tools -lisäosalla ja uudempi Android Studio. Tarjolla on lisäksi Android Developers -verkkosivusto, jolta löytyy tietoa kaikista Android-sovelluskehityksen vaiheista.

Haasteellista Android-sovelluskehityksessä on se, että käyttöjärjestelmä löytyy mitä moninaisimmilta laitteilta suurista tableteista pienimpiin älypuhelimiin. Kehityksessä tulee sen vuoksi ottaa huomioon niin näyttöjen koon vaihtelevuus kuin erot laitteiden suorituskyvyssä ja muistien määrissä. Niiden lisäksi on päätettävä, mitä versioita sovellus tukee. Android-puhelinten käyttöjärjestelmää ei ole pakko päivittää uusimpaan versioon, jonka vuoksi käyttäjillä saattaa olla käytössä hyvinkin vanhoja versioita Androidista (Skogberg 2010).

3.3 Android-sovellus

Android-sovellukset rakentuvat neljästä komponentista, joita ovat Activityt, Servicet, Content providerit ja Broadcast receiverit. Yksi Activity, eli aktiviteetti, vastaa sovelluksen kutakin näkymää. (Android 2015.) Aktiviteettien yhteydessä voidaan käyttää Fragment-komponentteja. Koska fragmentit ovat periaatteessa aliaktiviteetteja, on niitä käyttämällä mahdollista rakentaa näkymä, jolla on useita eri toimintoja yhtä aikaa. (Android 2015.)



Kuvio 4 Aktiviteetin elinkaari (Android 2015)

Kuviossa 4 on esitetty Android-aktiviteetin elinkaari. Sovelluksen käynnistyessä kutsutaan luotavan aktiviteetin onCreate-metodia, jonka jälkeen kutsuttavat metodit riippuvat käyttäjän toimista. On myös mahdollista, että sovellus ja sen kautta aktiviteetti tuhotaan muistinhallinnan vaatiessa lisäresursseja (Leukkunen 2012). Aktiviteetin elinkaari on otettava huomioon Android-sovelluksia kehitettäessä.

Valmiita Android-sovelluksia varten on olemassa virallinen jakoalusta Google Play Store. Sovellusten julkaiseminen Play Storessa vaatii maksullisen Google Developers -rekisteröitymisen. Rekisteröitymisen yhteydessä saa käyttöoikeuden Developer Console -sivustoon, jonka kautta sovelluksia voi julkaista myös Alpha- ja Beta-versioina.

4 TYÖSSÄ KÄYTETYT TEKNIIKAT

4.1 Ohjelmistot

Android Studio

Android Studio on toinen virallisista Android-sovelluskehitykseen käytettävissä olevista ohjelmointiympäristöistä. Android Studio valittiin sovelluksen ohjelmointiympäristöksi pääasiassa sen uutuuden takia. Työn aikana käytettiin versiota 1.1.0. Viimeisenä testauspäivänä versiosta 1.2.0 julkaistiin stabiili versio, jonka kaikkia uusia ominaisuuksia ei ehditty kokeilla.

Android Studio sisältää tehokkaita ominaisuuksia ja työkaluja. Kun ohjelmistoympäristössä esimerkiksi luodaan uusi projekti, kaikki tarvittavat tiedostot ja kansiot rakentuvat automaattisesti. Projektin muuntaminen tiedostoista toimivaksi Android-sovellukseksi onnistuu niin ikään yhdellä napin painalluksella. Tällaiset ominaisuudet helpottavat ohjelmistoprojektin hallintaa, jättäen näin enemmän aikaa itse ohjelmoinnille.

Sybase PowerDesigner

Sovelluksen teknisessä suunnittelussa käytettiin SAP:n Sybase PowerDesigner -ohjelmistoa. Ohjelmistolla on mahdollista mallintaa kaavioiden avulla melkein mitä tahansa. Sillä voi tehdä mm. UML-standardin mukaisia kaavioita, joiden avulla voidaan mallintaa ohjelmistoja ja tietokantoja. Tämän raportin kuvioiden 9, 10 ja 14 esittämät kaaviot on tehty PowerDesigneria käyttäen.

Fluid UI

Fluid UI on mobiilisovellusten prototypointityökalu, joka soveltuu erityisesti käyttöliittymäsuunnitteluun. Fluid UI on web-palvelu, joten sitä voi käyttää mistä vain. Palvelun vahvuuksia ovat sen laaja, yli 2000 elementtiä sisältävä käyttöliittymäelementtikirjasto ja mahdollisuus linkittää näkymiä toisiinsa (Fluid Software 2015). Näiden vahvuuksien takia palvelua päätettiin hyödyntää sovelluksen suunnittelussa.

4.2 Ohjelmointi- ja merkintäkielet

Java

Android Studiota käytettäessä sovellusten ohjelmointi tapahtuu Javalla, joka on korkean tason olio-ohjelmointikieli. Android-sovelluskehityksen yhteydessä käytettävässä Javassa on erityispiirteitä verrattuna perinteiseen, puhtaassa Java-ympäristössä tapahtuvaan Java-ohjelmointiin. Erot johtuvat Androidin Java-kirjaston ei-standardinmukaisuudesta ja Android-sovellusten elinkaaresta. (Leukkunen 2012.)

SQL

SQL on useimpien relaatiotietokantojen ainoa tuettu tietokantakieli. SQL-kieltä käytetään relaatiotietokannoissa olevien tietojen hakemiseen ja hallintaan. Sillä voi lisätä ja poistaa rivejä tietokannasta ja tehdä siihen päivityksiä. (Hovi ym. 2005, 10). Sovelluksessa SQL-kieltä käytetään SQLite-tietokannan hallintaan.

XML

XML (eXtensible Markup Language) on joustava ja yksinkertainen tekstin merkintätapa (W3C 2015). XML-tiedostoilla on suuri rooli Android-ohjelmistoprojekteissa. Esimerkiksi käyttöliittymän näkymien asettelutiedostot ovat XML-muotoisia. Lisäksi mm. sovellusten merkkijonolistojen arvot on tapana tallentaa projektin arrays.xml-tiedostoon. Koodiesimerkissä 1 on yksi sovelluksen merkkijonolistoista. Listan sisältö on kuvattu XML-merkinnöin siten, että merkkijonolista on merkitty merkillä string-array ja sen sisältämät merkkijonot merkein item.

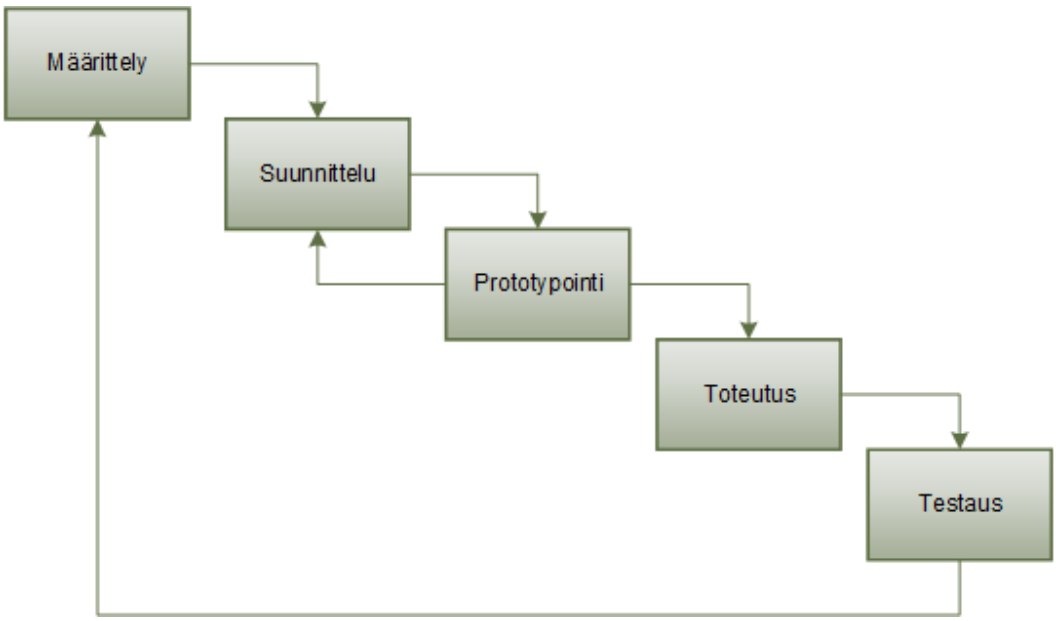
```
<string-array name="shapes_of_cat">
    <item>lean</item>
    <item>normal</item>
    <item>overweight</item>
</string-array>
```

Koodiesimerkki 1. Kissan vartalotyyppivaihtoehdot XML-merkinnöin

4.3 Ohjelmistokehityksen ja työnhallinnan mallit

4.3.1 Prototypoiva ohjelmistokehitys

Sovelluksen kehitykseen sovellettiin Prototype Driven Development (PDD) -kehitysmallia, joka voidaan kääntää prototypoivaksi ohjelmistokehitykseksi. Ohjelmistotuotannossa prototyypillä tarkoitetaan jollain tavoin vaillinaista järjestelmää, ohjelmaa tai sen osaa. Prototypoinnin avulla voidaan nopeasti muuttaa ideoita konkreettisiksi malleiksi, jolloin tuotteen käyttäjiltä on mahdollista saada välitöntä palautetta. Lisäksi prototypoinnin aikana voi punnita eri kehitysratkaisuja tuotteen varsinaista toteutusvaihetta varten. (Haikala & Mikkonen 2011; Kune & van Erkel 2003.)



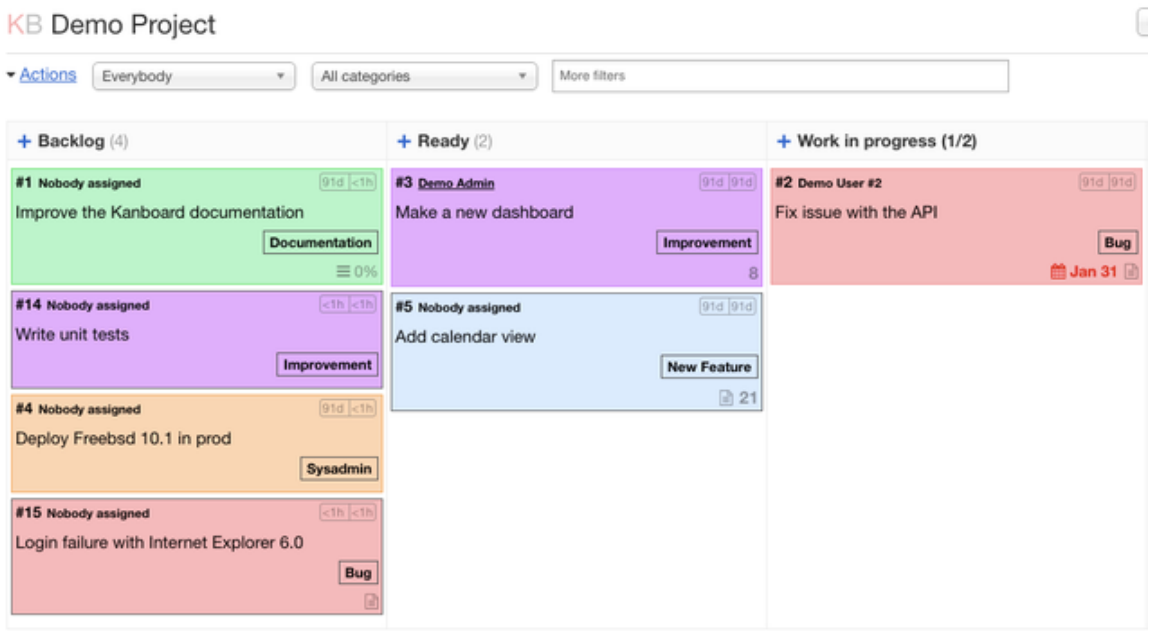
Kuvio 5. Prototypoivan iteraatiomallin vaiheet

Kuvio 5:n mukaista PDD-mallia päätettiin soveltaa, koska sovellusten kehitys Androidille ei ollut ennestään tuttua ja koska henkilökohtainen oppimistyyli on käytännöllinen. Prototyyppejä rakentamalla oli mahdollista yhtäaikaisesti opiskella Android-sovelluskehitystä käytännössä ja tarkentaa sovelluksen teknistä suunnitteludokumentaatiota. Kunen ja van Erkelin (2003) mukaan prototypointiprosessi onkin "iteratiivinen oppimisprosessi, jota kuvaavat toistuvat kehitys-testaus-oppimis-parannus-kierrokset."

4.3.2 Kanban

Työn hallintaan sovellettiin Kanban-metodia. Kanban perustuu työnkulun visualisointiin, yhtäaikaisten tehtävien rajoittamiseen ja kunkin tehtävän läpimenoajan mittaamiseen (Haikala & Mikkonen 2011, 55). Kanban otettiin käyttöön Kanboard-työkalun avulla.

Kuviossa 6 on nähtävissä esimerkki Kanban-taulun rakenteesta. Kanban-taulussa on yleensä neljä saraketta: backlog, ready, work in progress ja done. Sarakkeisiin sijoitetaan tehtäväkortteja. Kanboardissa kortteja voi helposti siirtää sarakkeesta toiseen vetämällä. Niiden lisääminen ja poistaminen on myös nopeaa.



Kuvio 6. Taulu Kanboard-työkalussa (Guillot 2015)

5 SOVELLUKSEN MÄÄRITTELY JA SUUNNITTELU

5.1 Esitutkimus

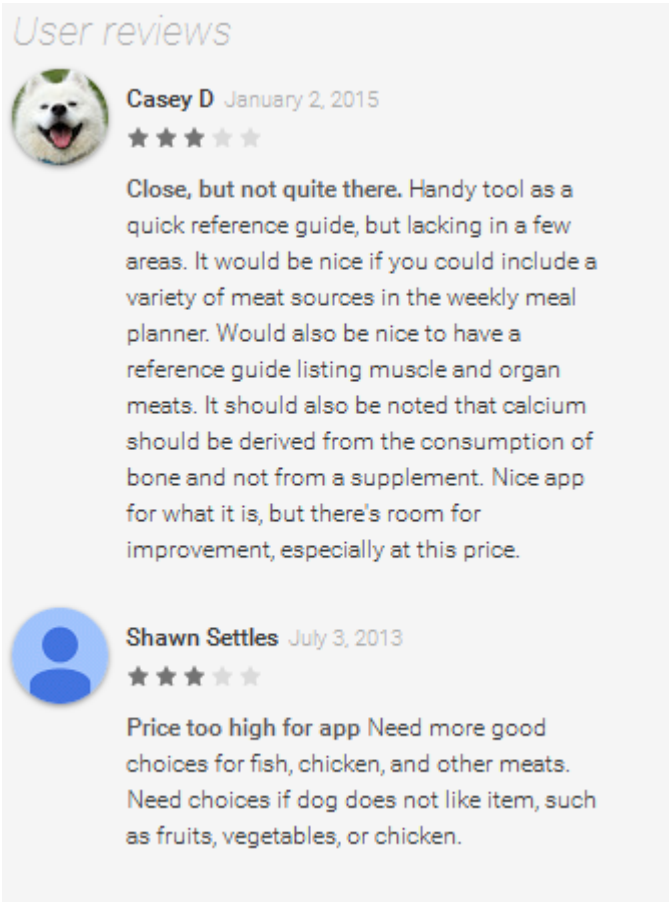
5.1.1 Tietojen kerääminen

Sovellusta varten kerättiin suuri määrä ravintoarvoja käyttäen Finelin (Terveyden ja hyvinvoinnin laitos 2013) ja USDA:n (2011) ravintoainetietokantoja. Tietokannoista otettiin vain raakojen ja valmistamattomien tuotteiden tiedot. Kun jonkin tuotteen ravintoarvot oli ilmoitettu molemmissa tietokannoissa, laskettiin niiden keskiarvo suuremman luotettavuuden saavuttamiseksi.

Lisäksi koottiin kissan energia- ja ravintoainetarpeet. Aineistoon sovellettiin lineaarista regressioanalyysiä, jolloin saatiin kuviossa 2 esitetyt yhtälöt. Näitä yhtälöitä käytetään sovelluksen ohjelmakoodissa kunkin kissatyypin päivittäisen energiantarpeen laskemiseen.

5.1.2 Markkina-analyysi

Jotta sovelluksesta saatava hyöty varmistuisi, analysoitiin jo saatavilla olevia sovelluksia Google Play Storessa. Hakusanoilla "cat" yhdistettyinä termeihin "feeding", "care", "nutrition" löytyi enimmäkseen pelejä, joissa kissoille syötetään leivonnaisia. Niiden joukossa oli myös sovelluksia, joiden toiminnot vastasivat sovelluksen tavoiteltua toiminnallisuutta. Samankaltaisia sovelluksia olivat ruokintalaskurit, jotka toimivat siten, että käyttäjä syöttää kissanruokapaketissa olevat tiedot sovellukseen, joka laskee kuinka suuri osa kaloreista tulee proteiinista, rasvasta ja hiilihydraateista.



Kuvio 7. Dog Raw Diet Calculatorin käyttäjäarvostelut (Google 2013)

Hakua laajennettiin vaihtamalla sana "cat" -sanaan "pet", jolloin löytyi koiranomistajille suunnattu maksullinen ruokintasovellus: Abcportalin kehittämä Dog Raw Diet Calculator. Kuviossa 7 nähtävien arvostelujen parannusehdotukset ja moitteet pyritään ottamaan huomioon sovelluksen kehityksessä.

5.2 Yleiskuvaus

Määriteltävä sovellus on Android-käyttöjärjestelmälle toteutettava kissojen raakaruokavalion suunnittelussa ja toteutuksessa avustava työkalu. Sovelluksen kohderyhmä on kissanomistajat, jotka ovat kiinnostuneita kissansa hyvinvoinnista ja ruokinnasta. Sovellus helpottaa kissanomistajan tehtäviä laskemalla ravintoaineet kissan viikoittaisessa ruokavaliossa ja antamalla mahdollisuuden verrata niitä kissan tarvitsemiin ravintoaineisiin. Palautetta ruokavaliosta antaa lisäksi kissan painon seurantatoiminto.

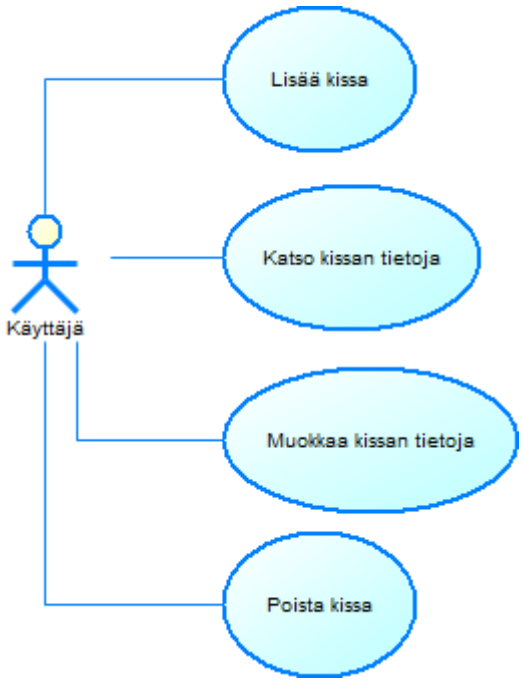
Sovellus tähdätään Androidin uusimmalle versiolle, joka on tällä hetkellä 5.1.1. Alin tuettu versio on 4.0.3. Toteutuksessa hyödynnetään XML-tiedostoja ja

SQLite-tietokantoja. Näytön orientaatioista tuetaan toistaiseksi vain pystyasentoa ja sovelluksen suunnittelussa jätetään vähemmälle huomiolle sovelluksen ulkoasu.

5.3 Toiminnot

5.3.1 Kissojen hallinta

Kuviossa 8 esitetään kissojen hallintaan liittyvät käyttötapaukset, joista ensimmäinen on kissan lisääminen. Käyttäjä pystyy tallentamaan sovellukseen kissansa tiedot, joita ovat nimi, rotu, syntymäpäivä, sukupuoli, paino, aktiivisuuden taso ja muoto. Lisäksi käyttäjä voi halutessaan lisätä sovellukseen kissan kuvan.



Kuvio 8. Kissojen hallinnan käyttötapaukset

Tallentamisen ehdot ovat seuraavan listan mukaiset:

- Kissalla on nimi.
- Kissan syntymäpäivä ei ole tulevaisuudessa.
- Kissan paino on positiivinen luku nollan ja 25:n välillä.
- Uroskissa ei ole kantava.
- Steriloitu naaraskissa ei ole kantava.

Tallentamisen jälkeen kissan tietoja pystyy muokkaamaan painon, aktiivisuuden tason ja muodon osalta. Tallennetun kissan tiedot voi poistaa, jota ennen sovellus pyytää vielä varmistuksen käyttäjältä. Kissan mukana poistetaan myös sen ruokalistat ja painohistoria.

Käyttäjän tallentamien kissojen kuvat ja nimet näkyvät sovelluksen aloitusnäkymässä. Jos kuvaa ei ole asetettu, näkyy sen sijaan vain kissan nimi. Nimipainiketta painamalla käyttäjä pääsee määrittelemään ja hallitsemaan kissan ruokalistoja ja käyttämään sovelluksen muita toimintoja.

5.3.2 Kissan painon seuraaminen

Raakaruokinnassa on hyvä seurata kissan yleiskuntoa ja painoa, jotta varmistutaan annetun ruoan sopivasta määrästä. Käyttäjän päivittäessä kissan painon muokkausnäkymässä tallentaa sovellus uuden painon muokkauspäivämäärineen tietokantaan. Käyttäjä voi tarkastella painon kehitystä siihen tarkoitetussa näkymässä.

Painon kehitys esitetään visuaalisesti kuvion 17 mukaisena käyränä lukuun ottamatta vihreää ennustekäyrää. Jos painoja on pitkältä aikaväliltä, käyttäjä pystyy valitsemaan näytetyn aikavälin. Oletusaikaväli on viimeiset kaksi kuukautta.

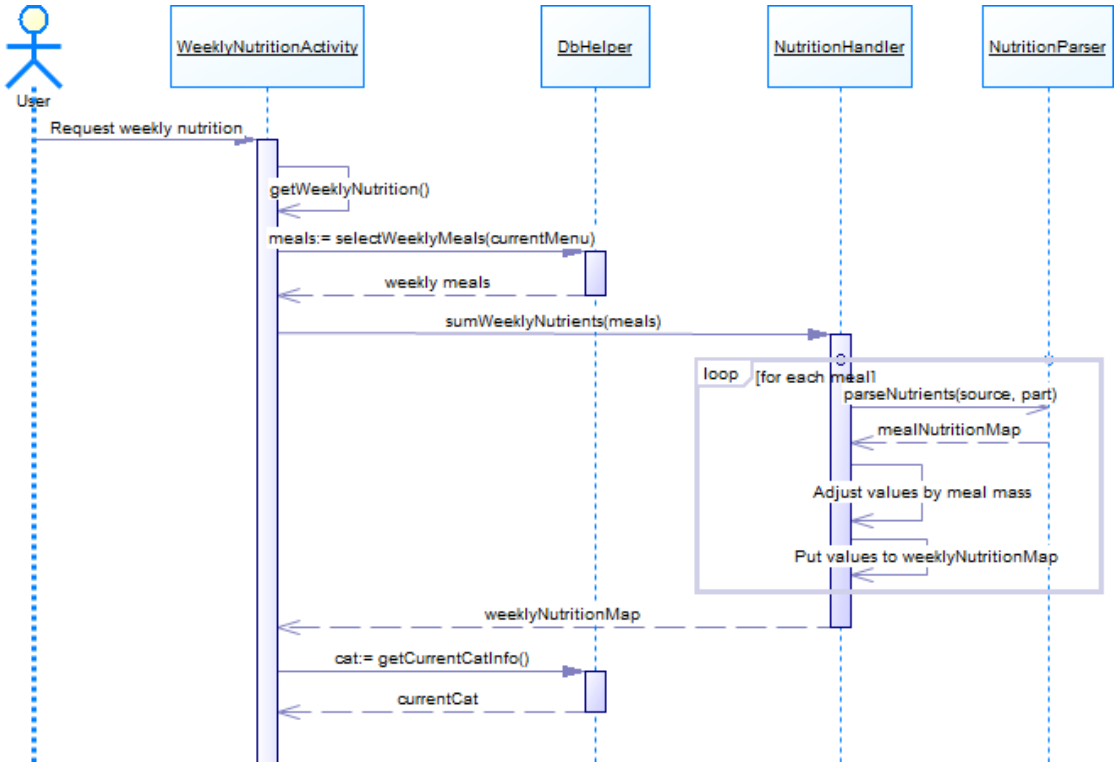
5.3.3 Ruokalistojen hallinta ja tarkastelu

Käyttäjä voi tallentaa kullekin kissalle viikoittaisia ruokalistoja, jotka voi tallentamisen jälkeen poistaa. Uusi ruokalista tallentuu järjestelmään, kun käyttäjä on lisännyt siihen ensimmäisen aterian. Ruokalistaan voi tallentaa viikon kullekin päivälle rajattomasti aterioita, joita voi myös muokata ja poistaa. Käyttäjä voi tarkastella ruokalistoja päivä kerrallaan, jolloin aterioiden tiedot listataan.

Ateriaa tallennettaessa valitaan ensin lihan lähde, joka voi olla esimerkiksi kalkkuna. Sen jälkeen valitaan kissalle syötettävä osa, joka voi kalkkunan tapauksessa olla esimerkiksi siipi. Aterialla on aina oltava positiivinen, enintään nelilukuinen paino grammoina. Lisäksi käyttäjä voi valita aterian ruokinta-ajan.

5.3.4 Ruokalistan ravintoarvojen katselu

Käyttäjä pystyy katselemaan ruokalistan aterioiden lisäksi sen ravintoarvoja ja vertaamaan niitä kissansa suositeltuihin ravintoarvoihin. Esitys tapahtuu visuaalisessa muodossa, jotta käyttäjän olisi helpompi havaita mahdolliset puutteet kissan ruokavaliossa. Visuaalinen muoto toteutetaan käyttäen vaakapylväskaavioita.



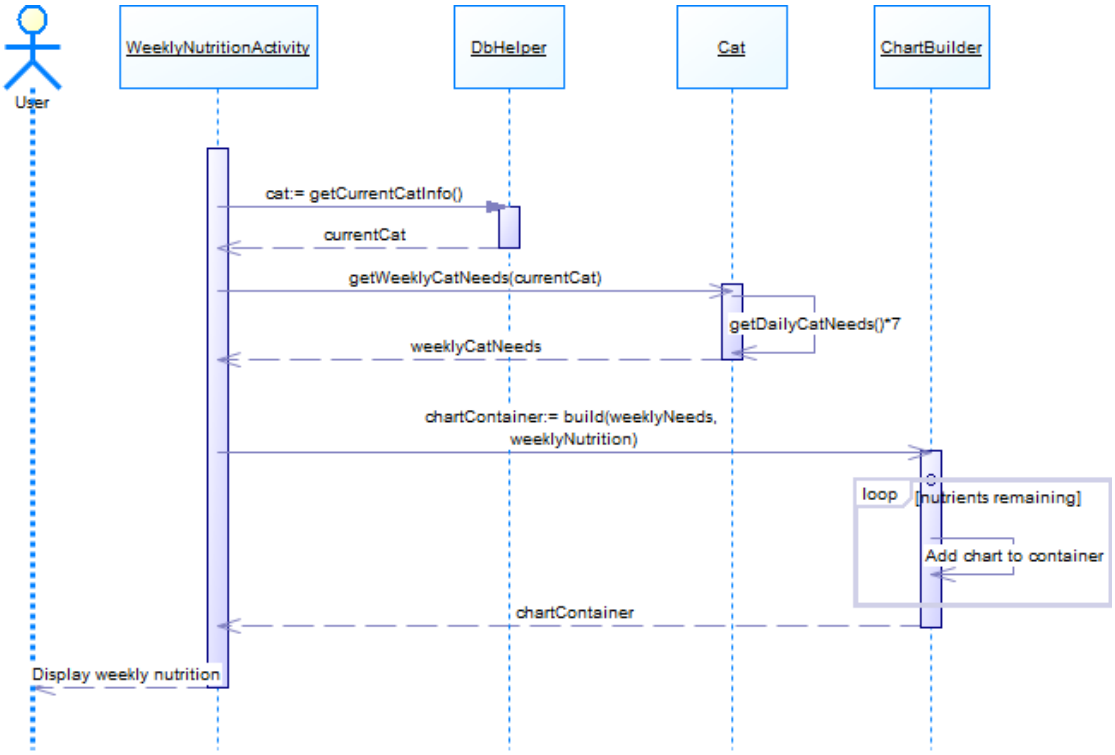
Kuvio 9. Ravintoarvojen vertailutoiminnon sekvenssikaavio, osa 1

Kuviossa 9 esitetään ravintoarvojen tarkastelun sekvenssikaavion ensimmäinen osa. Kaavion tilanteessa käyttäjä on painanut ravintoarvojen tarkastelunäkymään, WeeklyNutritionActivityyn, johtavaa painiketta. Activity pyytää SQLite-tietokannan käsittelijäluokalta DbHelperiltä valitun viikon ateriat. DbHelper hakee ateriat tietokannasta ja palauttaa ne Activitylle.

Ateriatiedot saatuaan Activity pyytää ravintoarvojen käsittelijäluokkaa NutritionHandleria laskemaan yhteen niiden ravintoarvot. Handler välittää pyynnön XML-tiedostojen parseriluokalle NutritionParserille ateria kerrallaan. Saatuaan ravintoarvot Handler säätää niitä suhteessa aterian painoon ja lisää ne viikoittaiseen

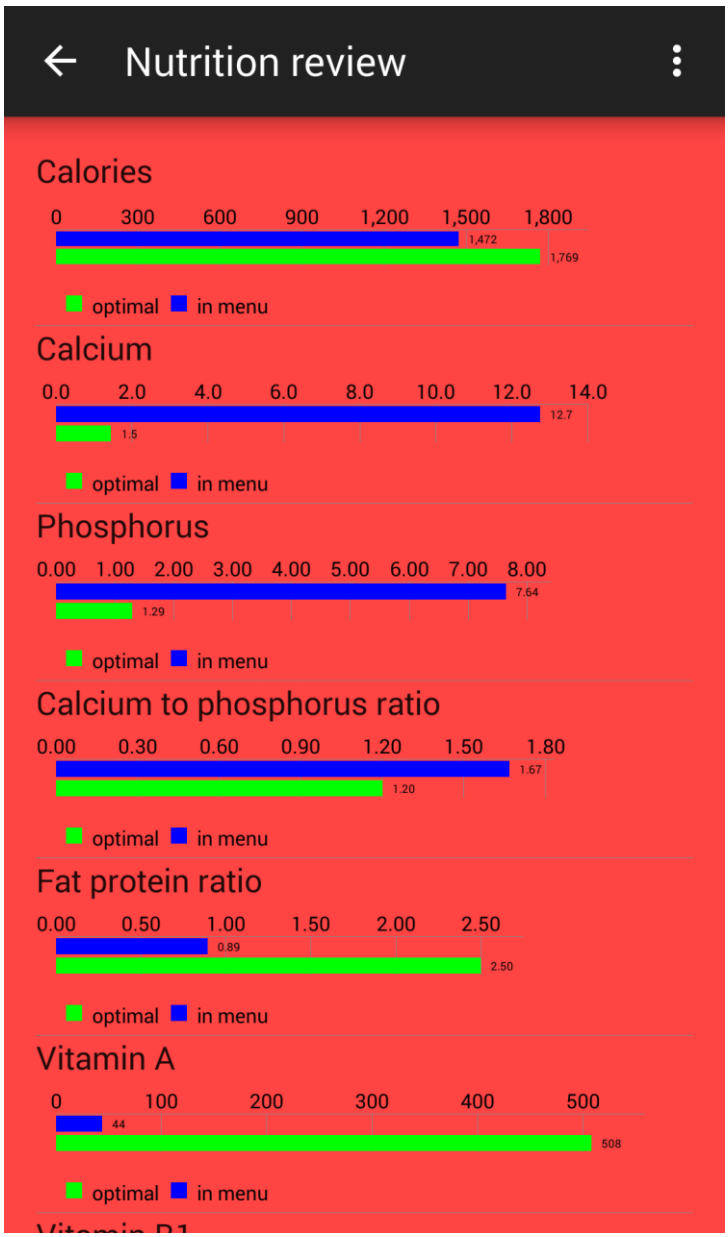
ravintoarvokokoelmaan. Kun kaikki ateriat on käsitelty, lähettää Handler kokoelman Activitylle.

Vertailua varten tarvitaan lisäksi kissan viikoittaiset ravintoainetarpeet. Kuviossa 10 esitetään sekvenssikaavion toinen osa. Activity pyytää kissan tiedot DbHelperiltä ja pyytää sitten Cat-luokkaa laskemaan kissan tarpeet. Cat palauttaa kokoelman tarpeista kerrottuaan päivittäiset tarpeet seitsemällä, joka on päivien määrä viikossa. Tämän jälkeen Activity välittää kaavioiden rakentajaluokalle ChartBuilder sekä ruokalistan ravintoarvot että kissan tarpeet. Builder rakentaa taulukon kustakin ravintoaineesta ja lisää ne chartContainer-objektiin, joka on osa näkymää. Lopulta Activity näyttää ravintoarvokaaviot käyttäjälle.



Kuvio 10. Ravintoarvojen vertailutoiminnon sekvenssikaavio, osa 2

Kuviossa 11 esitetään käyttäjälle näytetyt kaaviot. Kaavioissa on merkitty vihreällä suositeltujen ravintoaineiden arvojen palkit ja sinisellä ruokalistan sisältämien ravintoaineiden arvojen palkit. Palkkeja vertailemalla on helppo havaita mahdolliset puutteet tai yliannostuksen vaarat ruokalistassa.



Kuvio 11. Ravintoarvojen tarkastelunäkymä

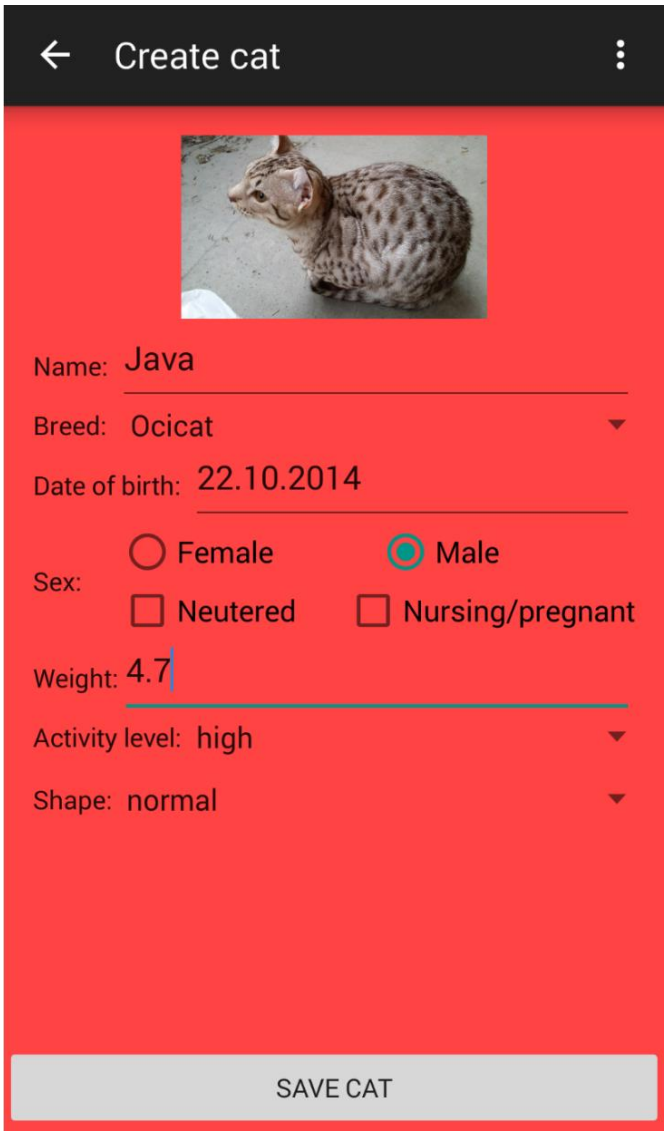
5.3.5 Käyttäjän tietojen synkronointi pilvipalveluun

Sovellus varmuuskopioi käyttäjän tallentamat tiedot pilvipalveluun, jotta ne eivät hukkuisi käyttäjän esimerkiksi kadottaessa laitteensa. Sovelluksen käyttäjäksi rekisteröityminen on edellytys tietojen varmuuskopioinnille. Rekisteröityminen tapahtuu sovelluksen sisällä, eikä se ole pakollista. Rekisteröityneen käyttäjän tarvitsee kirjautua sovellukseen sisään vain kerran.

Synkronointi pilvipalveluun tapahtuu automaattisesti laitteen ollessa yhteydessä Internetiin. Synkronointi tapahtuu sovelluksen taustalla omassa säikeessään.

Tällöin se ei hidasta sovelluksen käyttöliittymän säiettä ja siten vaikuta negatiivisesti käyttäjäkokemukseen. Kun käyttäjä kirjautuu sovellukseen sisään uudella laitteella, synkronointi pilvipalvelusta laitteeseen tapahtuu välittömästi, jolloin käyttäjää pyydetään odottamaan tietojen latautumista.

5.4 Käyttöliittymä- ja käytettävyyssuunnittelu



Kuvio 12. Kissan tietojen tallennusnäkymä

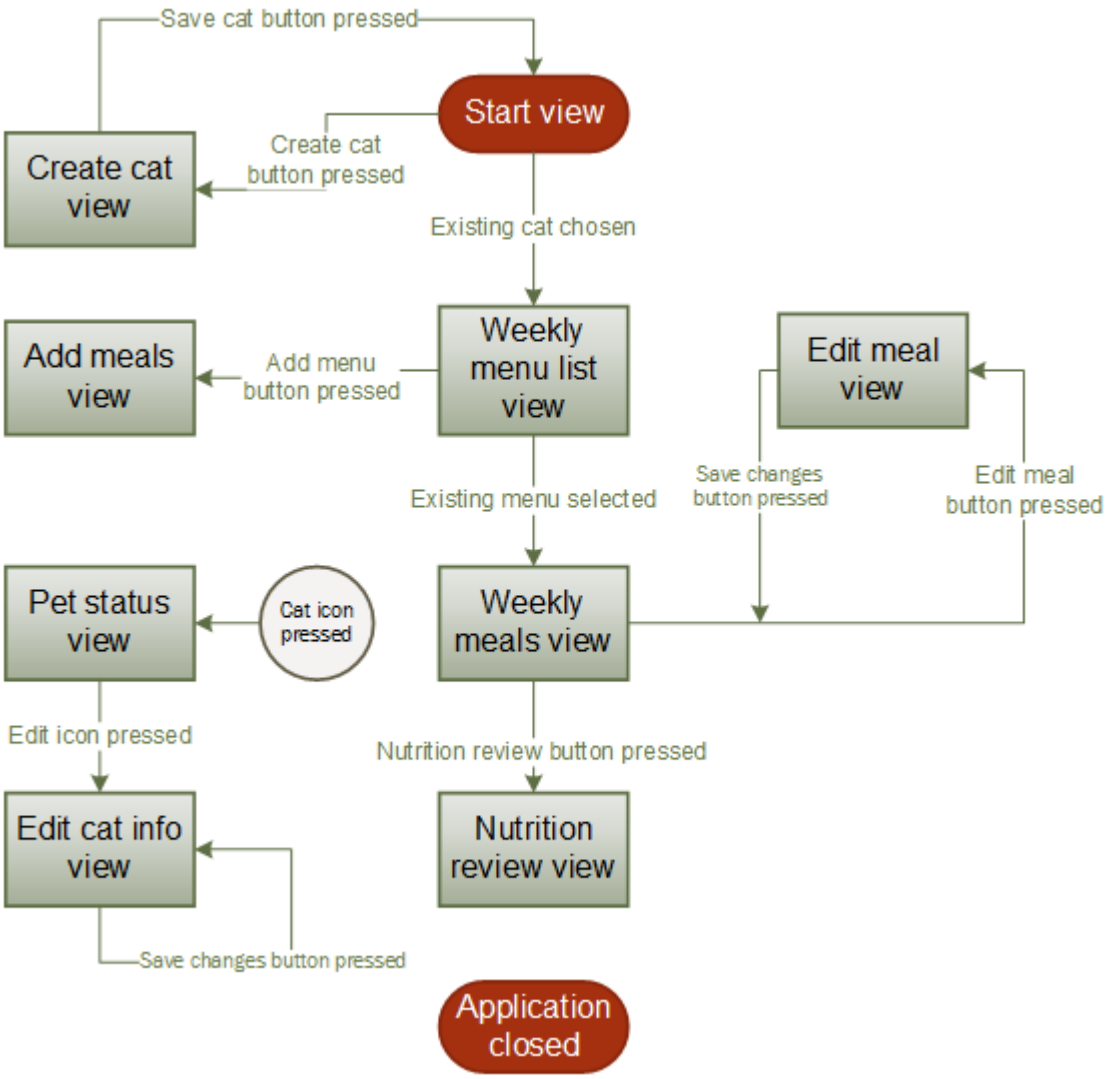
Sovelluksen suunnittelu tehtiin käyttäjälähtöisesti, mikä tarkoittaa että käyttäjän tarpeet otettiin huomioon suunnittelun kaikissa vaiheissa. Suunnittelu aloitettiin sen vuoksi käyttöliittymäsuunnittelulla, joka on hyvä tapa tarkastella toteutettavan sovelluksen käyttäjäystävällisyyttä jo ennen toteutusvaihetta. Kuviossa 12 esitetyssä kissan tietojen tallennusnäkymässä käytettävyyttä havainnollistaa se, että

käyttäjän sijainti sovelluksessa on nähtävissä Action Barissa. Jos käyttäjä syöttää näkymässä virheellisiä tietoja, tulee näkymään selkokielinen virheilmoitus.

Käyttöliittymäsuunnittelu aloitettiin hahmottelemalla sovelluksen näkymät käsin paperille. Piirrokset osoittautuivat tehokkaaksi keinoksi muovata näkymistä mielekkäitä hahmotelmia siten, että kukin näkymä piirrettiin yhä uudelleen kunnes siitä ei löytynyt enää parannettavaa. Piirrokset siirrettiin tämän jälkeen sähköiseen muotoon Fluid UI -palvelussa, jossa näkymät linkitettiin toisiinsa. Näin saatiin aikaan sovelluksen alustava näkymäkokonaisuus.

Navigaatiomallin suunnitteluvaiheessa pyrittiin ottamaan huomioon tämän hetken design guidelinet. Alun perin sovelluksen navigoinnin oli pääasiassa tarkoitus tapahtua Action Barissa olevasta Dropdown-elementistä ja käyttää tabbed-välilehtiä, mutta API-taso 21:n myötä kyseiset navigaatiomallit olivat vanhentuneet (Android 2014), mikä tarkoitti, että navigaatiolle oli etsittävä vaihtoehto. Google suosittelee kehittäjiä käyttämään navigaatioelementtinä Navigation Draweria, joka päätettiin ottaa käyttöön sovelluksessa.

Kuviossa 13 esitetään sovelluksen tämän hetkinen navigaatiomalli. Laitteen back-painiketta painamalla pääsee edelliseen näkymään, Action Barissa oleva Up-painike vie käyttäjän ylöspäin näkymähierarkiassa, joka on määritelty sovelluksen Manifestissa. Kaaviossa näkyvä Cat icon pressed -elementti viittaa useimpien näkymien Action Barissa sijaitsevaan kissaikoniin, joka vie käyttäjän kissan tiedot -näkymään.

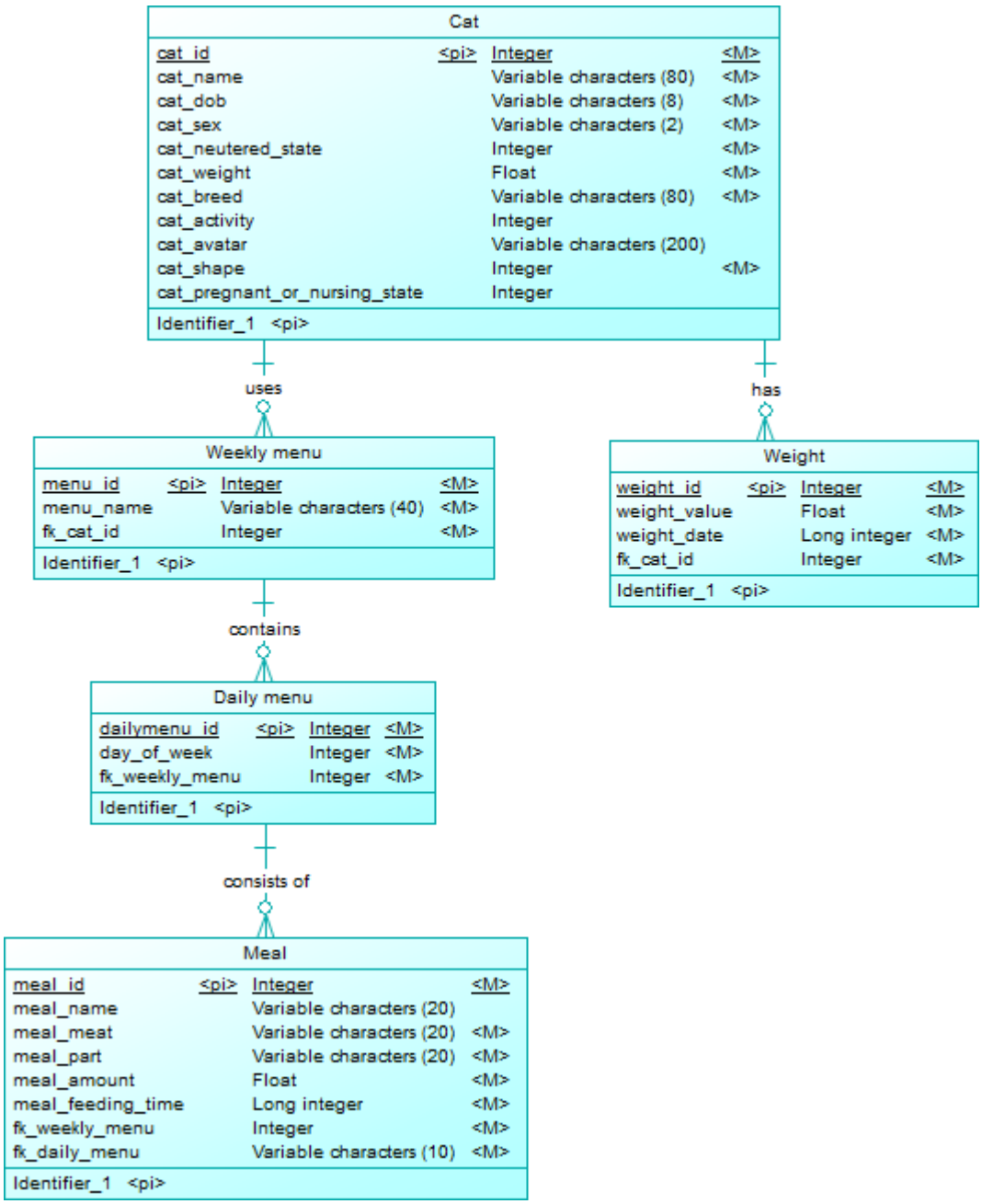


Kuvio 13. Sovelluksen navigaatiomalli

5.5 Tietokantasuunnittelu

5.5.1 Käyttäjän syötteet

Käyttäjän syötteiden tallennusta varten suunniteltiin kuvion 14 CDM-kaavion mukainen SQLite-tietokanta. Kaavio kuvaa samalla sovelluksen POJO-luokkien Meal, Cat ja WeeklyMenu jäsenmuuttujat, jotka ovat samat kuin kaavion taulujen kentät avaimia lukuunottamatta. Kaikissa tauluissa on uniikki perusavain id, jota käytetään haettaessa tietokannasta taulujen rivejä. Päätaulun Cat perusavaimeen viitataan taulujen Weekly menu ja Weight foreign key -avaimissa. Taulut Daily menu ja Meal viittaavat taulun Cat riveihin Weekly menun kautta, jolloin kaikki taulut ovat yhteydessä päätauluun.



Kuvio 14. CDM-kaavio sovelluksen tietokannasta

5.5.2 Ravintoainetietokanta

Ravintoaineiden tallennusta varten olisi ollut myös mahdollista käyttää SQLite-tyyppistä tietokantaa, mutta tiedot päädyttiin sen sijaan tallentamaan XML-tiedostoon. XML-tiedostoja voi muokata lähes millä tahansa tekstinkäsittelyohjelmalla, mikä tekee ravintoainetietojen ylläpidon ja muokkaamisen helpoksi. Ne ovat lisäksi helppolukuisia ja siirrettäviä.

```
<food>
    <chicken>
        <part>
            <neck>
                <common_values>
                    <calories>297</calories>
                    <carbohydrates>0</carbohydrates>
                    <fat>26</fat>
                    <protein>14</protein>
                    <bone>40</bone>
                    <meat>60</meat>
                    <calcium>18</calcium>
                    <phosphorus>112</phosphorus>
                    <moisture>59.99</moisture>
                    <ash>0.55</ash>
                </common_values>
                <vitamins...>
                <other_values...>
            </neck>
            <back...>
            <wing...>
        </part>
    </chicken>
    <turkey...>
</food>
```

Kuvio 15. Ravintoaine-XML-tiedoston rakenne

Kuvio 15 on nähtävissä tiedoston rakenne. Juurielementin food sisällä ravinnonlähteet ovat omina elementteinään, joiden sisällä on niiden syötävät osat, kuten kanojen kaulat ja siivet. Osaelementtien sisältä löytyvät itse ravintoarvot, jotka on jaoteltu selkeyden vuoksi omiin ryhmiinsä. Vitamins-elementin sisällä ovat vitamiinit, other_values-elementti sisältää mineraalit ja tarkemmat rasvahappoerittelyt.

6 TOTEUTUS

6.1 Toteutus yleisesti

Ennen kutakin iteraatiota suunniteltiin, mitä suunnitelluista toiminnoista ja ominaisuuksista iteraatiossa aletaan toteuttamaan. Ominaisuuksien toteutus jaettiin sopivan kokoisiksi tehtäviksi, jotka lisättiin Kanboardin backlog-sarakkeeseen. Tehtäviä oli samanaikaisesti käynnissä enintään viisi. Kunkin vaiheen lopussa suoritettiin koodin katselmointi ja refaktorointi ja sovellusta testattiin laitteella.

Toteutuksen aikana ilmenneiden ongelmien ratkaisussa käytettiin apuna kysymys-vastaus -palstaa Stack Overflow. Yhden henkilön voimin sovellusta toteutettaessa ei verkkopohjaista versionhallintaa, kuten Gittiä, koettu tarpeelliseksi konfiguroida. Sen sijaan versionhallinta toteutettiin tallentamalla projektikansio päivämäärineen pilvipalveluun päivittäin ja aina ennen suurien muutosten tekoa.

6.2 Ensimmäinen iteraatio

Ensimmäisen iteraation tavoite oli selvittää, ovatko suunnitelmat toteutettavissa sellaisenaan Android-alustalla. Iteraatiota varten sovelluksen toiminnot ja ominaisuudet asetettiin tärkeysjärjestykseen. Koska sovelluksen perimmäinen tavoite on toimia työkaluna raakaruokavalion suunnittelussa, korkeimman prioriteetin toiminnot olivat kissan tallennus järjestelmään, sen valinta aloitusnäkymästä ja valitun aterian parserointi.

Iteraation aikana toteutettiin prototyyppitason toiminnot korkean prioriteetin toiminnoista. Yksi prototyyppitoiminnoista oli ennalta määrätyn aterian kalorien parserointi XML-tiedostosta ja niiden näyttäminen käyttäjälle. Koodiesimerkissä 2 esitetään kyseisen toiminnon toteutus kooditasolla.

```
    mEditGrams.addTextChangedListener(new TextWatcher()
    {
        @Override
        public void onTextChanged(CharSequence s, int start, int before, int
count)
        {
            try
            {
                if (count > 0 || start > 0)
                {
                    grams = Float.parseFloat(s.toString());
```

```
                    displayCalories(grams);
                }
                else
                    caloriesView.setText("");

            } catch (Exception e) {}
        }
    });
}

private void displayCalories(float grams)
{
    float m = NutritionHandler.calculateMultiplier(grams);
    caloriesView.setText("Calories: " + m * currentCalories);
}
```

Koodiesimerkki 2. Kaloreiden päivittäminen näkymään

Koodiesimerkin 2 koodissa mEditGrams viittaa käyttöliittymän aterioiden lisäysnäkymän painotekstikenttään. Tekstikentälle asetetaan koodissa TextWatcher-tyyppinen kuuntelija, jossa metodia onTextChanged kutsutaan aina kun käyttäjä muuttaa tekstikentän tekstiä. If-lauseen ehto on, että muuttuneen tekstin pituus on suurempi kuin nolla. Ehto täyttyy myös, jos teksti alkaa muuttuneen syötteen kohdasta nolla. Jos ehto ei täyty kentän muututtua tyhjäksi, tyhjennetään myös caloriesView-elementin teksti.

Jos ehto täyttyy, yritetään parseroida tekstikentän muuttunut sisältö ja tallentaa se float-tyyppiseen grams-olioon. Parserointi johtaa poikkeukseen, jos se ei onnistu. Jos parserointi onnistuu, kutsutaan displayCalories-metodia. Metodissa pyydetään aterian painosta laskettu säätöluku. CaloriesView-kentässä näytetään sen jälkeen säätöluvulla säädetyt kalorit.

6.3 Toinen iteraatio

Sovelluksen toisen iteraation aikana käytettiin hyväksi ensimmäisen iteraation aikana tarkentuneita suunnitelmia. Toisen iteraation tavoitteena oli laajentaa toiminnallisuutta seuraavan listan mukaisiin toimintoihin:

- kissojen hallinta
- ruokalistojen hallinta ja tarkastelu
- ruokalistan ravintoarvojen vertailu suosituksiin.

Ensimmäisestä versiosta puuttuva kissojen ja ruokalistojen poistamistoiminto lisättiin koodiin iteraation aikana. Koodiesimerkissä 3 esitetään tietojen poistamista edeltävän käyttäjän varmistuksen pyytämisen toteutus. Ominaisuus toteutettiin rakentamalla Dialog-tyyppinen luokka. Dialogin onCreateDialog-metodia kutsutaan, kun se luodaan Activityssä.

```
public Dialog onCreateDialog(Bundle savedInstanceState)
{
    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

    builder.setMessage("Are you sure you want to delete?").setPositiveBut-
ton("Yes", new DialogInterface.OnClickListener()
    {
        @Override
        public void onClick(DialogInterface dialogInterface, int i)
        {
            mListener.onDialogPositiveClick(ConfirmDeleteDialogFragment.this);
        }
    })
            .setNegativeButton("Cancel", new DialogInterface.OnClickListener()
            {
                public void onClick(DialogInterface dialog, int id)
                {
                    mListener.onDialogNegativeClick(ConfirmDeleteDialogFrag-
ment.this);
                }
            });

    return builder.create();
}
```

Koodiesimerkki 3. Tietojen poistamisen varmennustoiminnon toteutus

OnCreateDialogissa käytetään AlertDialog.Builder-tyyppistä luokkaa rakentamaan Dialog. Dialogissa näkyväksi viestiksi asetetaan "Are you sure you want to delete?". Dialogin-kyllä painikkeen painaminen kutsuu kuuntelijaolion mListenerin onDialogPositiveClick-metodia, ei-painike taas johtaa onDialogNegativeClick metodiin. OnDialogPositiveClick-metodin seurauksena kissan tiedot poistetaan.

Virheilmoitusten näyttämistä varten rakennettiin poikkeusluokka InvalidInputException, jonka rakentajametodi on nähtävissä koodiesimerkissä 4. Rakentajametodille välitetään parametrina poikkeuksen syy, jonka perusteella määräytyy käyttäjälle näytettävän viestin, eli message-merkkijonon sisältö.

```
public InvalidInputException(int cause)
{
    switch(cause)
    {
        case INVALID_DATE_OF_BIRTH:
            message = "Please enter a valid date of birth";
```

```
            break;
        case INVALID_MEAL_AMOUNT:
            message = "Please enter a valid amount";
            break;
        case INVALID_WEIGHT:
            message = "Please enter a valid weight";
            break;
        case NAME_EMPTY:
            message = "Please enter a name";
            break;
    }
}
```

Koodiesimerkki 4. Poikkeusluokka InvalidInputExceptionin rakentajametodi

```
if (catName.equalsIgnoreCase(""))
    throw new InvalidInputException(InvalidInputException.NAME_EMPTY);
```

Koodiesimerkki 5. InvalidInputException-poikkeuksen heitto

```
catch (Exception e)
{
    String errorText = e.getMessage();
    Toast toast = Toast.makeText(getApplicationContext(), errorText,
Toast.LENGTH_SHORT);
    toast.show();
}
```

Koodiesimerkki 6. Poikkeuksen sieppaaminen ja näyttäminen käyttäjälle

Koodiesimerkissä 5 esitetään, kuinka poikkeus heitetään tyhjän kissan nimen tapauksessa kissan tallennusaktiviteetissa. Heittäminen tapahtuu try-catch-rakenteen try-osan sisällä, jolloin koodiesimerkin 6 catch-osa käsittelee poikkeuksen. ErrorText-merkkijonon sisältö on tässä tapauksessa "Please enter a name". Toast-luokan makeText-metodin avulla merkkijono asetetaan toast-ilmoituksen sisällöksi, joka lopulta näytetään käyttäjälle show-metodilla.

6.4 Kolmas iteraatio

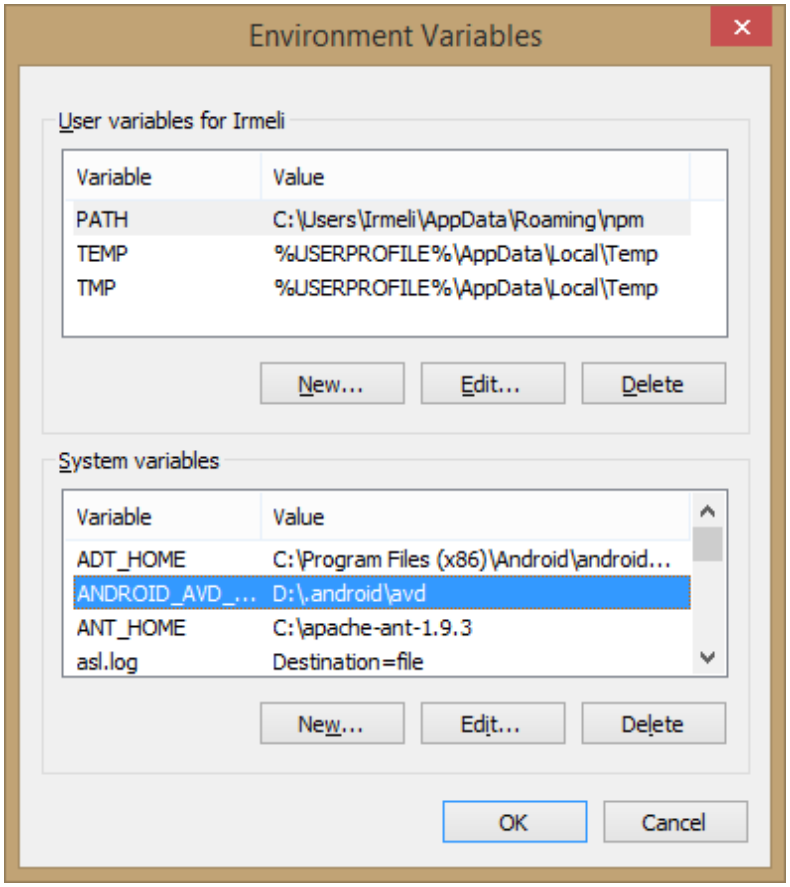
Kolmannen iteraation tavoitteena oli lisätä sovellukseen vaiheittain prototypoiden seuraavan listan mukaiset ominaisuudet:

- käyttäjän sisäänkirjautuminen
- käyttäjän tallentamien tietojen synkronointi Azure-palveluun
- käyttäjän tietojen haku Azure-tietokannasta laitteelle
- kissan painon seuraustoiminto
- navigaatiomallin uudistaminen.

Kirjautumis- ja synkronointiominaisuuksia varten luotiin Microsoftin Azure-pilvipalvelussa SQL-tietokanta. Palvelinpuolen autentikaatio rakennetiin alustavasti .NET-ympäristössä. Lisäksi sovellukselle hankittiin Facebook App ID Facebookin kautta kirjautumista varten. Navigaatiomallin muuttamiseksi alettiin muokata Activityistä Fragmentteja.

6.5 Ongelmakohtia

Sovelluksen debuggausta ja yksikkötestausta hankaloitti kaksi asiaa, joista ensimmäinen oli se että AVD:t eivät suostuneet käynnistymään. Android Studion ilmoittama virhe oli seuraavanlainen: "*PANIC: HOME is defined but could not find Nexus_5_API_21_x86.ini file in $HOME\.android\avd".* Virheen korjaaminen edellytti Windowsin Environment Variablesin muokkaamista kuvion 16 mukaisesti.



Kuvio 16. Environment Variables -ikkuna

Toinen ongelma debuggauksessa oli Android Studion lokityökalun Logcatin toimimattomuus. Debuggauksessa on paljon hyötyä sovelluksen kaatumisilmoituksista, mutta Logcatiin ei tullut minkäänlaisia viestejä. Stackoverflow-tutkimuksen jälkeen selvisi, että kyse on Logcatissa ajoittain ilmenevästä ohjelmistovirheestä, joka korjaantuu joko käynnistämällä työkalu tai koko ohjelmointiympäristö uudestaan (Android Open Source Project 2014).

7 TESTAUS

Testaussuunnitelma

Sovelluksen manuaalinen testaus tehtiin käyttäen Samsungin GT-I9506-älypuhelinta, jossa oli asennettuna Androidin versio 4.4.2. Testaus suoritettiin lisäksi kahdella virtuaalisella laitteella, joista ensimmäisen Android-versio oli Ice Cream Sandwich (4.0.3), eli sovelluksen alin tuettu versio. Toisen laitteen versio oli Lollipop (5.1.1), joka on tällä hetkellä Androidin uusin ja siten sovelluksen ylin tuettu versio. Testitapaukset määräytyivät sovelluksen vaatimusmäärittelyn mukaisesti. Testitapausten lisäksi sovellus yritettiin saada kaatumaan käyttäen sitä niin sanotusti väärin kaikin mahdollisin tavoin.

Testauksen tulokset

Sovelluksesta ei löytynyt suuria virheitä testauksen aikana. Virheet olivat oletettavasti karsiutuneet toteutuksen aikaisen yksikkötestauksen ja debuggauksen seurauksena. Joitakin pieniä virheitä oli jäänyt testausta edeltäneestä koodin refaktoroinnista, joiden korjaaminen johti uusiin virheisiin. Lopulta nekin saatiin korjattua, jonka seurauksena syntyi toimiva sovellus.

Alkuperäisen testaussuunnitelman mukaan sovellus oli tarkoitus testata myös automaattisesti, mutta ajanpuutteen vuoksi sitä ei ehditty tekemään opinnäytetyön aikana. Sen vuoksi esimerkiksi tuhannen aterian lisäyksen vaikutusta sovelluksen toimintaan ei päästy testaamaan ja on mahdollista, että virheitä on vielä olemassa.

8 TULOKSET JA JATKOKEHITYS

8.1 Tulokset

Opinnäytetyön tuloksena syntyi Android-sovellus, joka on käytettävissä kissan raakaruokavalion suunnitteluun ja toteutukseen. Sovelluksen käyttäjä voi lisätä sovellukseen kissansa tiedot ja hallita niitä. Käyttäjä voi tehdä viikoittaisia ruokalistoja, joihin voi kullekin päivälle lisätä aterioita. Ateriat ovat muokattavissa ja poistettavissa. Ruokalistojen ravintoarvoja voi verrata kissan yksilöllisiin suositusravintoarvoihin visuaalisessa muodossa. Sovelluksen alemman prioriteetin toimintojen toteutus jätettiin jatkokehitykseen.

Sovelluksen laskemien ravintoarvojen luotettavuutta heikentää se, että ravintoainetietokannoissa ilmoitetuissa arvoissa on puutteita. Esimerkiksi useimpien raaka-aineiden B-vitamiiniarvot eivät olleet lainkaan saatavilla. Suurin puute oli kuitenkin se, että luiden ravintoaineita ei oteta tietokannoissa lainkaan huomioon, mikä oletettavasti johtuu siitä, etteivät ihmiset usein syö luita. Sen vuoksi luita sisältävien raaka-aineiden näytetyt mineraalimäärät ovat pienempiä kuin ne todellisuudessa ovat.

Prototypoivan ohjelmistokehitysmallin soveltaminen osoittautui erinomaiseksi keinoksi opiskella Android-ohjelmointia samalla projektia eteenpäin vieden. Mallin tehokkuuden tekee kuitenkin kyseenalaiseksi laatuerot toteutuksen alku- ja loppupuolen ohjelmakoodeissa. Sovellus rakentui alustavan koodin päälle ja arkkitehtuurin muuttaminen jälkeenpäin oli aikaa vievää. Toisaalta voidaan pohtia, olisiko erilliseen opiskeluun käytetty aika ollut johtanut lyhempään kokonaisaikaan kuin koodimuutoksien teko.

8.2 Jatkokehitys

Sovelluksella on monia jatkokehitysmahdollisuuksia. Kehitystä jatketaan ensin toteuttamalla loppuun kolmannen iteraation suunnitellut ja keskeneräiset ominaisuudet, jonka jälkeen hiotaan käyttöliittymä miellyttävämmäksi ja näyttävämmäksi. Sovellus kaipaa lisää grafiikkaa: mm. ikonit kustakin eläimestä ja osasta raaka-ainevalikoissa parantaisivat käyttäjäkokemusta.

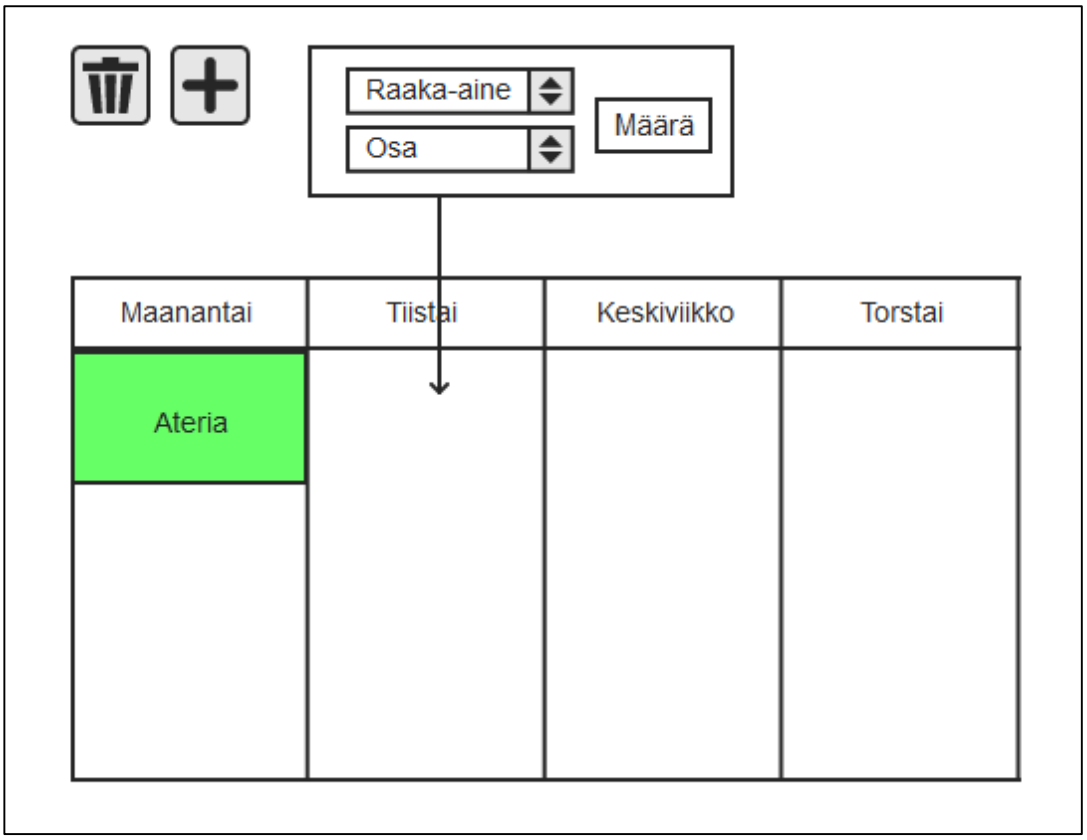
Tärkeää on myös parantaa sovelluksen ilmoittamien ravintoarvojen luotettavuutta, mikä edellyttää useampien ravintoainetietokantojen tutkimista. Jotta voidaan näyttää luita sisältävien ruoka-aineiden todelliset mineraalimäärät, on tutkittava kuinka suuri osa kustakin raaka-aineesta on luuta keskimääräisesti. On myös selvitettävä, kuinka suuren osan luusta kissa kykenee hyödyntämään.

Suositeltujen ravintoarvojen tarkentamiseksi on myös suunniteltu eri kissarotujen metabolisten erojen huomioon ottamista. Tietojen kerääminen rotujen edustajien painoista eri elinvaiheissa antaisi mahdollisuuden näyttää käyttäjälle ennusteen kissan painon kehityksestä kuvion 17 mukaisesti. Kaaviossa voisi lisäksi näyttää rodun keskimääräiset minimi- ja maksimipainot, jolloin kissan ideaalinen paino olisi helppo havaita.

Kuvio 17. Kissan painon kehityksen esitysmuoto

Määrittelyvaiheen markkina-analyysin aikana selvisi, että koirien raakaruokinnan suosiosta huolimatta samankaltaisten koiranomistajille kohdistettujen sovellusten tarjonta Play Storessa on vähäistä. Tarjonnan vähäisyyden ja oletettavasti suuren kysynnän vuoksi on aloitettu sovelluksen koiranomistajille kohdistetun version määrittely ja esitutkimus. Tällä hetkellä käynnissä on kasviperäisten raaka-aineiden ravintoarvojen kerääminen. Koirat nimittäin ovat kaikkiruokaisia, ja niille tulee syöttää lihan ja luiden ohessa myös kasviksia.

Lisäksi sovelluksen määrittelyn aikana potentiaaliset käyttäjät ilmaisivat kiinnostuksensa selaimella käytettävää web-palvelua kohtaan. Sovelluksen selainpohjainen toteutus avaisi päätelaitteiden suurempien näyttökokojen kautta uusia mahdollisuuksia.



Kuvio 18. Näkemys web-palvelusta

Yksi web-palvelun käyttöliittymän toteutusmahdollisuus on esitetty kuviossa 18. Kanban-tauluun perustuvassa näkymässä ateriat lisätään syöttämällä niiden tiedot pieneen laatikkoon, joka vedetään hiirellä halutulle päivälle. Samassa näkymässä on mahdollista näyttää välittömästi viikon ravintoarvot; päivittäiset arvot näkee valitsemalla halutun päivän. Aterioita voi niin ikään tarkastella tarkemmin ja muokata valitsemalla ne. Ateriat voi poistaa vetämällä ne roskakoriin.

9 YHTEENVETO

Opinnäytetyön tekeminen oli minulle mieluisaa, vaikka en kokenut Android-sovelluskehitystä yhtä luontevaksi kuin työskentelyä muilla mobiilialustoilla. Perinteisenä Java-ohjelmoijana siirtyminen Androidin kehitysympäristöön vaati tuttujen ajattelutapojen muuttamista. Tämän vuoksi Androidin laajaa ohjelmistokehystä ei onnistuttu sovelluksessa hyödyntämään erityisen tehokkaasti.

Työn vaiheet eivät pysyneet niille suunnitellussa aikataulussa. Jos tekisin työn uudelleen, jättäisin määrittelyvaiheen alustavasti suppeaksi ja siirtyisin nopeasti käyttöliittymäsuunnitteluun ja prototypointiin. Määrittelydokumentin kirjoittaminen oli prosessissa kaikista vaikeinta ja vei eniten aikaa, koska tyhjän paperin kammo ja perfektionismi vallitsivat. Suurin osa työstä tehtiin lopulta intensiivisten neljän viikon aikana.

Kaiken kaikkiaan opinnäytetyö on ollut todella mielenkiintoinen ja opettava prosessi. Se oli kuin koko insinöörikoulutuksen kulminoituma, jonka aikana ymmärsin kaikkien käymieni opintojaksojen merkityksen ja sain mahdollisuuden soveltaa käytännössä niiden aikana oppimiani asioita. Raakaruokinnan puolestapuhujana motivaatiota riitti, ja nyt minulla onkin työkalu kissojeni raakaruokavalion suunnitteluun.

LÄHTEET

Android Open Source Project. 2014. Google Project Hosting - Issue 80871. Viitattu 13.4.2015 https://code.google.com/p/android/issues/detail?id=80871.

Android. 2014. Android API Differences Report. Viitattu 1.5.2015 https://developer.android.com/sdk/api_diff/21/changes.html.

Android. 2015. Application Fundamentals. Viitattu 30.4.2015 http://developer.android.com/guide/components/fundamentals.html.

Android. 2015. Fragments. Viitattu 4.5.2015 http://developer.android.com/guide/components/fragments.html.

Android, 2015. Pausing and Resuming an Activity. Viitattu 5.5.2015 developer.android.com/training/basics/activity-lifecycle/pausing.html.

Antell, A. 2007. Kissan hoito. Viitattu 15.4.2015 http://www.fourpawdrive.fi/index.html%3Fpage_id=42.html.

Fluid Software. 2015. Fluid UI Features. Viitattu 4.5.2015 https://www.fluidui.com/features/.

Gartner. 2014. Gartner Says Sales of Smartphones Grew 20 Percent in Third Quarter of 2014. Viitattu 22.4.2015 http://www.gartner.com/newsroom/id/2944819.

Google. 2008. The first Android-powered phone. Viitattu 25.4.2015 http://googleblog.blogspot.fi/2008/09/first-android-powered-phone.html.

Google. 2013. Dog Raw Diet Calculator. Viitattu 29.4.2015 https://play.google.com/store/apps/details?id=com.petdietexperts.dograw-dietcalculator.full.

Guillot, F. 2015. Kanboard is a simple visual task board software. Viitattu 29.4.2015 http://kanboard.net/screenshots/tour/board.png.

Haikala, I. & Mikkonen, T., 2011. Ohjelmistotuotannon käytännöt. 12., uudistettu painos. Hämeenlinna: Talentum Media Oy.

Hovi, A., Huotari, J. & Lahdenmäki, T. 2005. Tietokantojen suunnittelu & indeksointi. 1. painos. Porvoo: Docendo.

Itkonen, J. 2012. Ohjelmistoarkkitehtuurit 2012. Viitattu 4.5.2015 http://users.jyu.fi/~ji/opetus/oa2012/luento-14.html.

Kune, H. & van Erkel, F. 2003. Rapid Prototyping. Weesp: Educore.

Leukkunen, A. 2012. Java-ohjelmoinnin erityispiirteet Android-pohjaisissa laitteissa. Tietojenkäsittelytieteiden laitos. Jyväskylän yliopisto. Kandidaatintutkielma.

Lipponen, P. 2012. Kissan luonnonmukainen ruokinta. Viitattu 16.4.2015 http://pikkupeto.blogspot.fi/2012/12/kissan-luonnonmukainen-ruokinta.html.

Liu, J. & Yu, J. 2011. Research on Development of Android Applications. Kun-Ming: IEEE.

Multanen, A., 2013. Artikkeli kissojen raakaruokinnasta. Viitattu 28.4.2015 http://www.annamultanen.net/artikkeli-kissojen_luonnonmukainen_ruokinta.html.

National Research Council. 2006. Your cat's nutritional needs. Washington: National Academies Press.

Nederhoff, T. 2012. Droog versus Rauw. Petfood Magazine 2012:3, 18-19.

Open Handset Alliance, 2012. Open Handset Alliance. Viitattu 22.4.2015 http://www.openhandsetalliance.com/.

Pierson, L. 2013. Feeding Your Cat: Know the Basics of Feline Nutrition. Viitattu 30.4.2015 http://www.catinfo.org/.

Skogberg, B. 2010. Android Application Development. Tietojenkäsittelytieteiden laitos. Malmön yliopisto. Opinnäytetyö.

Terveyden ja hyvinvoinnin laitos, 2013. Elintarvikkeiden koostumustietopankki. Viitattu 29.4.2015 http://www.fineli.fi.

USDA, 2011. USDA National Nutrient Database for Standard Reference. Viitattu 9.4.2015 http://ndb.nal.usda.gov/.

W3C, 2015. Extensible Markup Language. Viitattu 2.5.2015. http://www.w3.org/XML/.

Webopedia. 2014. What is POJO? Viitattu 4.5.2015 http://www.webopedia.com/TERM/P/POJO.html.

Xamarin Inc. 2015. Working with AndroidManifest.xml. Viitattu 5.5.2015 http://developer.xamarin.com/guides/android/advanced_topics/working_with_androidmanifest.xml/.

YSA. 2010. VESA - Verkkosanasto. Viitattu 30.4.2015 http://vesa.lib.helsinki.fi/ysa/.

Zoran, D. 2002. The carnivore connection to nutrition in cats. Journal of the American Veterinary Medical Association 2002:11, 1559.